IB893

Intel® Pineview D
ICH8 Chipset
3.5” Disk Size SBC

USER’S MANUAL

Version 1.0

Acknowledgments

AMI BIOS is a trademark of American Megatrends Inc.
PS/2 is a trademark of International Business Machines Corporation.
Intel is a registered trademark of Intel Corporation.
Microsoft Windows is a registered trademark of Microsoft Corporation.
Winbond is a registered trademark of Winbond Electronics Corporation.
All other product names or trademarks are properties of their respective owners.

Table of Contents

This page is intentionally left blank.

Introduction

Product Description

The IB893 is a 3.5-inch disk-size SBC featuring the Intel® Atom™ DC D525 processor with ICH8M chipset. It has one DDR3-800MHz SO-DIMM memory slot that can fit in up to 4GB of system memory. VGA graphics support dual 24-bit LVDS interface.

The board has edge connectors for VGA CRT, two RJ45, two LAN/USB stack connector and a serial port. A total of four RS232 serial ports are available on board. Expansion is provided with a Mini PCI-e socket.

Other features include two SATA connectors, digital I/O, KB/mouse headers, watchdog timer, and a +12V DC-in power connector. Board dimensions are 102mm and 147mm.

Checklist

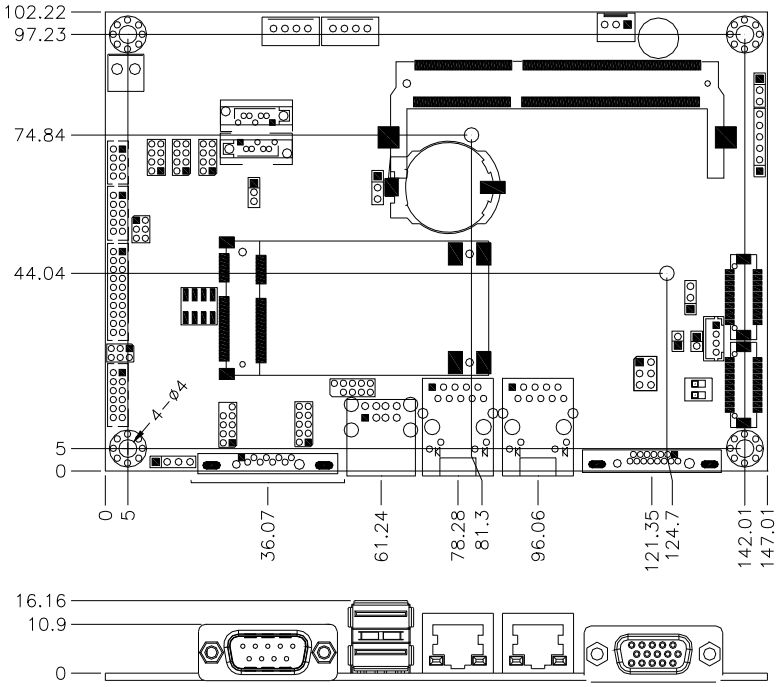
Your IB893 package should include the items listed below.

- The IB893 3.5” disk-size SBC
- This User’s Manual
- 1 CD containing chipset drivers and flash memory utility

IB893 Specifications

Product Name	IB893
Form Factor	3.5”
CPU Type	Intel® Pineview-D DC Processor
CPU Speed	Intel® Atom™ DC D525 / 1MB cache/1.66GHz (13W) 22mm x 22mm, Micro-FCBGA8
BIOS	AMI BIOS
Chipset	ICH8M: 31mm x 31mm,T-PBGA (2.4W)
Memory	DDR3-800MHz SO-DIMM x 1 (w/o ECC), Max. 4GB , Single channel
VGA	Intel® Integrated Graphics Controller Supports DirectX 9 Graphic (200MHz/ 400MHz), OpenGL 1.4
LVDS	Xilinx XC3S200AN-4FTG256C FPGA for converting 18-bit to 24-bit Support 24-bit dual channels LVDS interface w/DF13 socket x2
LAN	Realtek 8111G-CG x 2 (32-pin QFN package)
USB	ICH8M built-in USB 2.0 host controller, support 8 ports
Serial ATA Ports	ICH8M built-in SATA controller, supports 2 ports
Audio	Intel ICH8M built-in audio controller w/ Realtek ALC662-VD Codec (supports 5.1 CH audio & SSM2304 2W/4ohm Amplifier
LPC I/O	Nuvoton NCT6627UD: COM1 (RS232/422/485) [EXAR SP339EER1 232/422/485 transceiver x 1 for jumper-less] COM2~COM4 (RS232 only) with pin-9 with power for 2 ports (500 mA for each port) **EXAR SP3243EBER x 3** Hardware monitor (2 thermal inputs, 4 voltage monitor inputs, VID0-4 & 1 x Fan Header)
Digital IO	4 in & 4 out
Keyboard/Mouse Connector	Thru pin-headers
Expansion Slots	Mini PCI-e socket x 1 (w/ USB signal)
Edge Connector	DB15 x1 for VGA RJ45 x2 for LAN 1&2 USB stack connector x 1 for USB1 ~2 DB9 x 1 for COM1
On Board Header/Connector	DF11 2x4-pin socket for KB/mouse 2x4-pin header x 3 for 6 USB ports DF13 socket connector x 2 for 24-bit dual channel LVDS 2x6-pin box header x1 for Audio 4-pin header x1 for speaker 2x5-pin box header x 1 for COM2 2x10-pin box header x 1 for COM3 ~4 2x5-pin headers x 1 for LPC (80-port card debugging purpose) Mini PCI-e(1x) connector x 1 4-pin box header x 1 for backlight/brightness control 4-pin power connector x 2 for SATA HDD
Watchdog Timer	Yes (256 segments, 0, 1, 2…255 sec/min)
Power Connector	+12V DC-in
RoHS	Yes
Board Size	102mm x 147mm

Board Dimensions



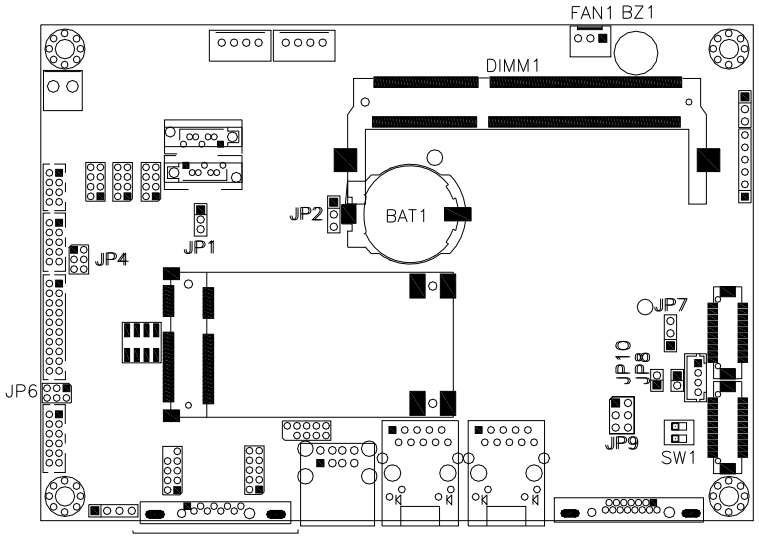
Installations

This section provides information on how to use the jumpers and connectors on the IB893 in order to set up a workable system. The topics covered are:

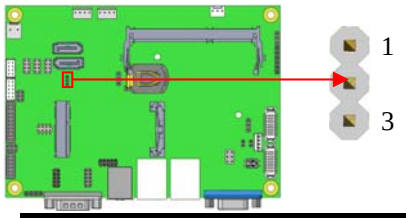
Setting the Jumpers

Jumpers are used on IB893 to select various settings and features according to your needs and applications. Contact your supplier if you have doubts about the best configuration for your needs. The following lists the connectors on IB893 and their respective functions.

Jumper Locations on IB893

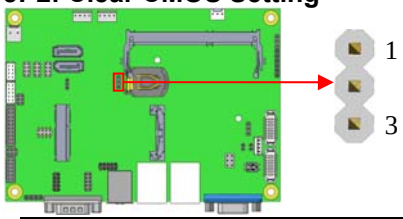


JP1: ATX/AT Mode Select



JP1	ATX / AT
	ATX mode
	AT mode

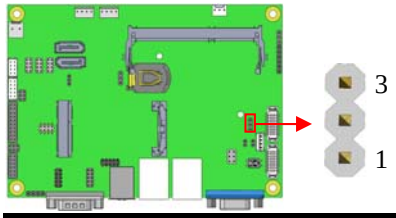
JP2: Clear CMOS Setting



JP2	Setting
	Normal
	Clear CMOS

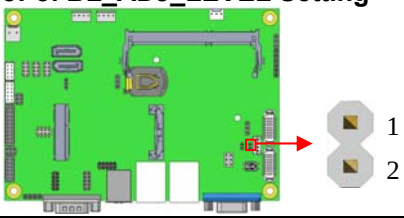
Note: Please remove the lithium battery before setting the jumper.

JP7: LCD Panel Power Selection



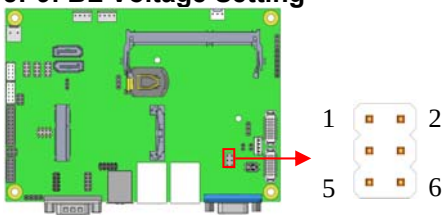
JP7	LCD Panel Power
	3.3V
	5V

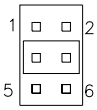
JP8: BL_ADJ_LEVEL Setting



JP8	Function
Open	3.3V
Close	5V (default)

JP9: BL Voltage Setting

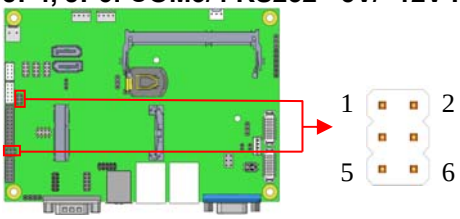


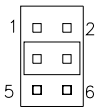
JP9	Setting	Function
	Pin 1-2 Short/Closed	+3.3V
	Pin 3-4 Short/Closed	+5V
	Pin 5-6 Short/Closed	+12V(Default)

JP10: BL_ADJ_Mode Setting

JP10	Function
Open	DC
Close	PWM

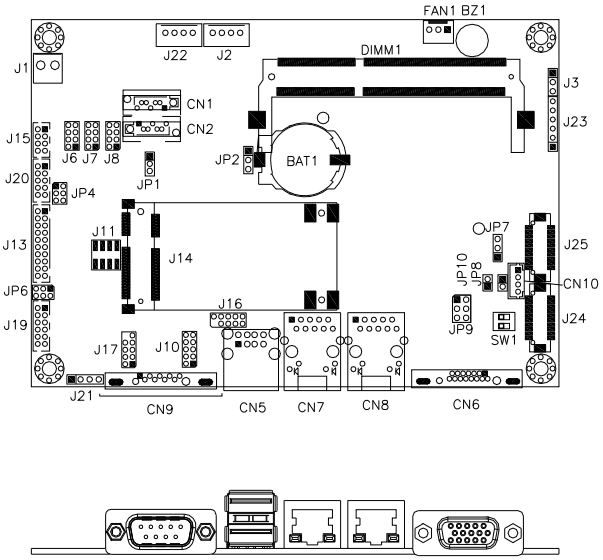
JP4, JP6: COM3/4 RS232 +5V/+12V Power Setting

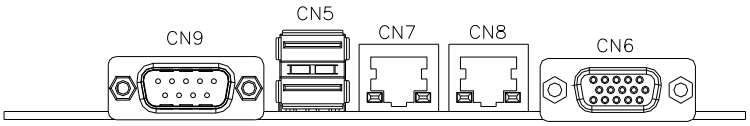


JP4/JP6	Setting	Function
	Pin 1-3 Short/Closed	+12V
	Pin 3-4 Short/Closed	Normal
	Pin 3-5 Short/Closed	+5V

Connectors on IB893

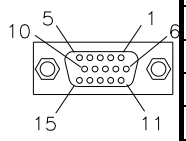
Connector Locations on IB893





CN5: USB 1/2 Connector

CN6: VGA Connector

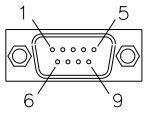


VGA

Signal Name	Pin #	Pin #	Signal Name
Red	1	2	Green
Blue	3	4	N.C.
GND	5	6	GND
GND	7	8	GND
VCC	9	10	GND
N.C.	11	12	DDCDATA
HSYNC	13	14	VSYNC
DDCCLK	15		

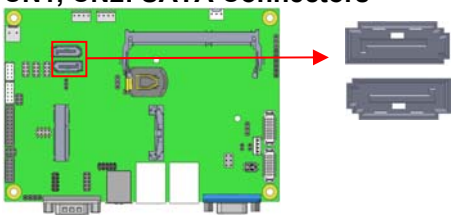
CN7, CN8: Gigabit LAN RJ45 Connector

CN9: COM1 RS232/RS422/RS485 Connector

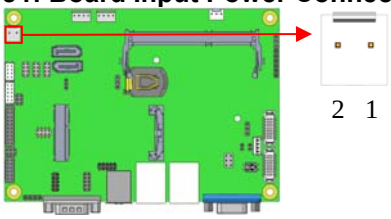


Pin #	Signal Name		
	RS-232	**R2-422**	**RS-485**
1	DCD	TX-	DATA-
2	RX	TX+	DATA+
3	TX	RX+	NC
4	DTR	RX-	NC
5	Ground	Ground	Ground
6	DSR	NC	NC
7	RTS	NC	NC
8	CTS	NC	NC
9	RI	NC	NC
10	NC	NC	NC

CN1, CN2: SATA Connectors

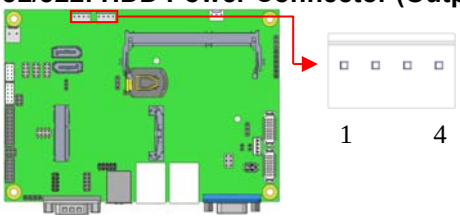


J1: Board Input Power Connector



Pin #	Signal Name
1	+12V
2	GND

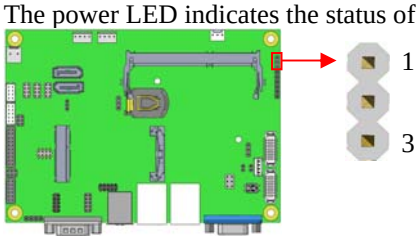
J2/J22: HDD Power Connector (Output Only)



Pin #	Signal Name
1	+5V
2	Ground
3	Ground
4	+12V

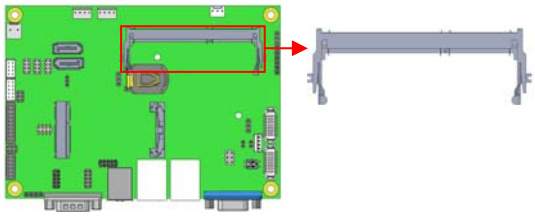
J3: Power LED

The power LED indicates the status of the main power switch.

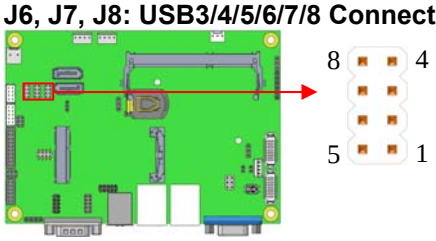


Pin #	Signal Name
1	Power LED
2	No connect
3	Ground

DIMM1: DDR3 SO-DIMM

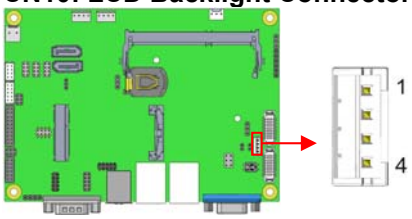


J6, J7, J8: USB3/4/5/6/7/8 Connector



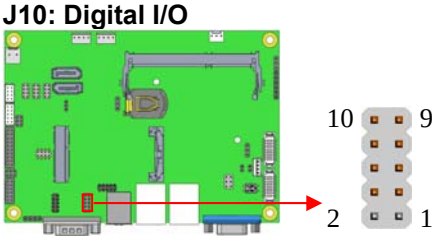
Signal Name	Pin #	Pin #	Signal Name
Vcc	1	5	Ground
D-	2	6	D+
D+	3	7	D-
Ground	4	8	Vcc

CN10: LCD Backlight Connector



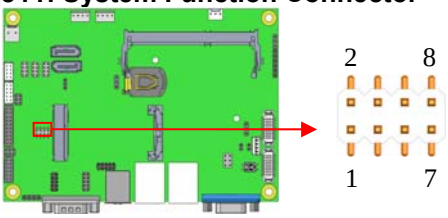
Pin #	Signal Name
1	+12V
2	Backlight Enable
3	Brightness Control
4	Ground

J10: Digital I/O



Signal Name	Pin #	Pin #	Signal Name
GND	1	2	VCC
OUT3	3	4	OUT1
OUT2	5	6	OUT0
IN3	7	8	IN1
IN2	9	10	IN0

J11: System Function Connector



Pin 1/2 ATX Power On Switch
Pin 3/4 HDD LED connector
Pin 5/6 Reset Switch
Pin 7/8 +5V and 5VSB signals

ATX Power ON Switch: Pins 1 and 2
This 2-pin connector is an "ATX Power Supply On/Off Switch" on the system that connects to the power switch on the case. When pressed, the power switch will force the system to power on. When pressed again, it will force the system to power off.

Hard Disk Drive LED Connector: Pins 3 and 4
This connector connects to the hard drive activity LED on control panel. This LED will flash when the HDD is being accessed.

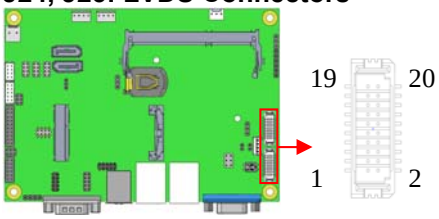
Pin #	Signal Name
4	HDD Active
3	5V

Reset Switch: Pins 5 and 6
The reset switch allows the user to reset the system without turning the main power switch off and then on again. Orientation is not required when making a connection to this header.

+5V and 5VSB Signals: Pins 7 and 8

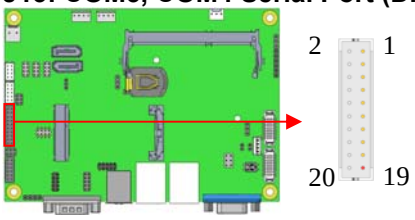
Pin #	Signal Name
7	+5V
8	+5VSB

J24, J25: LVDS Connectors



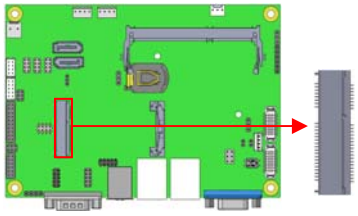
Signal Name	Pin #	Pin #	Signal Name
GND	19	20	N.C
ENABLE	17	18	LCD_PWR
CLK+	15	16	CLK-
GND	13	14	GND
LD2+	11	12	LD2-
LD3+	9	10	LD3-
GND	7	8	LCD_PWR
LD1+	5	6	LD1-
GND	3	4	GND
LD0+	1	2	LD0-

J13: COM3, COM4 Serial Port (DF11 Connector)

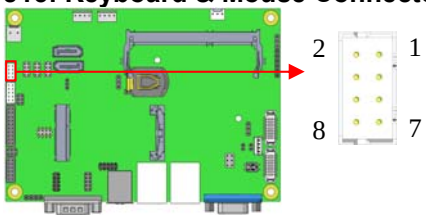


Signal Name	Pin #	Pin #	Signal Name
DSR3	2	1	DCD3
RTS3	4	3	RXD3
CTS3	6	5	TXD3
RI3	8	7	DTR3
NC	10	9	Ground
DSR4	12	11	DCD4
RTS4	14	13	RXD4
CTS4	16	15	TXD4
RI4	18	17	DTR4
NC	20	19	Ground

J14: Mini PCIE Connector

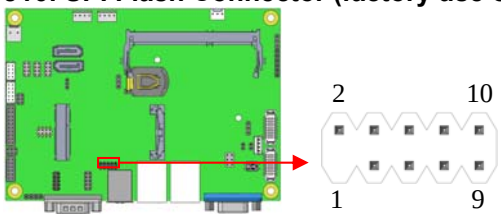


J15: Keyboard & Mouse Connector (DF11 Connector)

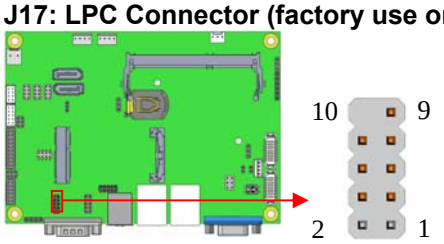


Signal Name	Pin #	Pin #	Signal Name
VCC	1	2	VCC
MDA	3	4	KBDA
MCL	5	6	KBCL
Ground	7	8	Ground

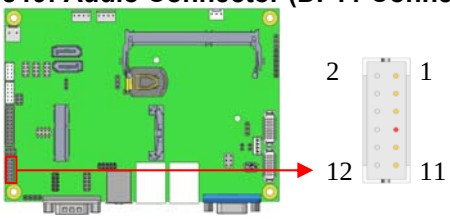
J16: SPI Flash Connector (factory use only)



J17: LPC Connector (factory use only)

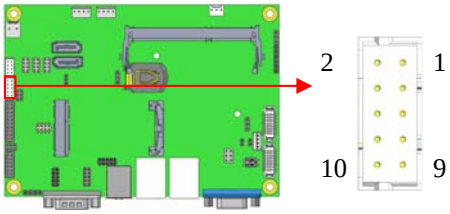


J19: Audio Connector (DF11 Connector)



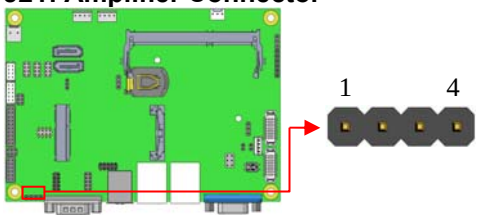
Signal Name	Pin #	Pin #	Signal Name
LINEOUT R	2	1	LINEOUT L
Ground	4	3	JD FRONT
LINEIN R	6	5	LINEIN L
Ground	8	7	JD LINEIN
MIC-In	10	9	MIC L
Ground	12	11	JD MIC1

J20: COM2/RS232 Serial Port



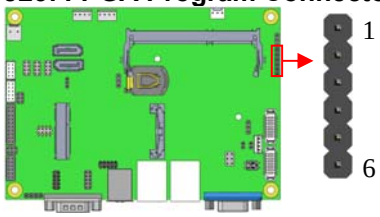
Signal Name	Pin #	Pin #	Signal Name
DCD, Data carrier detect	1	2	RXD, Receive data
TXD, Transmit data	3	4	DTR, Data terminal ready
GND, ground	5	6	DSR, Data set ready
RTS, Request to send	7	8	CTS, Clear to send
RI, Ring indicator	9	10	Not Used

J21: Amplifier Connector



Pin #	Signal Name
1	OUTL+
2	OUTL-
3	OUTR-
4	OUTR+

J23: FPGA Program Connector



LVDS Setting (Bios\Advance\GPIO Configuration\ setting)

GPIO	LVDS Type		
	18Bit Single	24Bit Single	24Bit Dual
12	High	High	High
17	High	Low	High
18	Low	High	High

BIOS SETUP

This chapter describes the different settings available in the AMI (American Megatrends, Inc.) BIOS that comes with the board. The topics covered in this chapter are as follows:

BIOS Introduction

The BIOS (Basic Input/Output System) installed in your computer system's ROM supports Intel processors. The BIOS provides critical low-level support for a standard device such as disk drives, serial ports and parallel ports. It also adds virus and password protection as well as special support for detailed fine-tuning of the chipset controlling the entire system.

BIOS Setup

The BIOS provides a Setup utility program for specifying the system configurations and settings. The BIOS ROM of the system stores the Setup utility. When you turn on the computer, the BIOS is immediately activated. Pressing the <Del> key immediately allows you to enter the Setup utility. If you are a little bit late pressing the <Del> key, POST (Power On Self Test) will continue with its test routines, thus preventing you from invoking the Setup. If you still wish to enter Setup, restart the system by pressing the "Reset" button or simultaneously pressing the <Ctrl>, <Alt> and <Delete> keys. You can also restart by turning the system Off and back On again. The following message will appear on the screen:

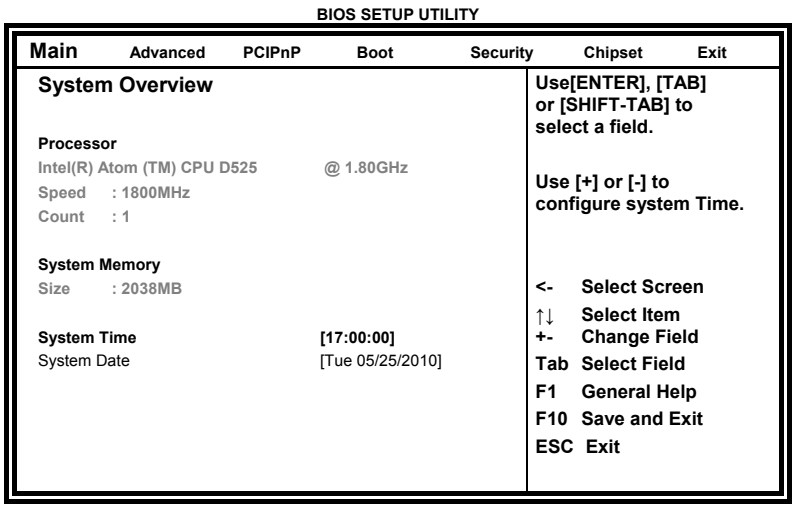
```
Press  <DEL>  to  Enter  Setup
```

In general, you press the arrow keys to highlight items, <Enter> to select, the <PgUp> and <PgDn> keys to change entries, <F1> for help and <Esc> to quit.

When you enter the Setup utility, the Main Menu screen will appear on the screen. The Main Menu allows you to select from various setup functions and exit choices.

Main BIOS Setup

This setup allows you to record some basic hardware configurations in your computer system and set the system clock.

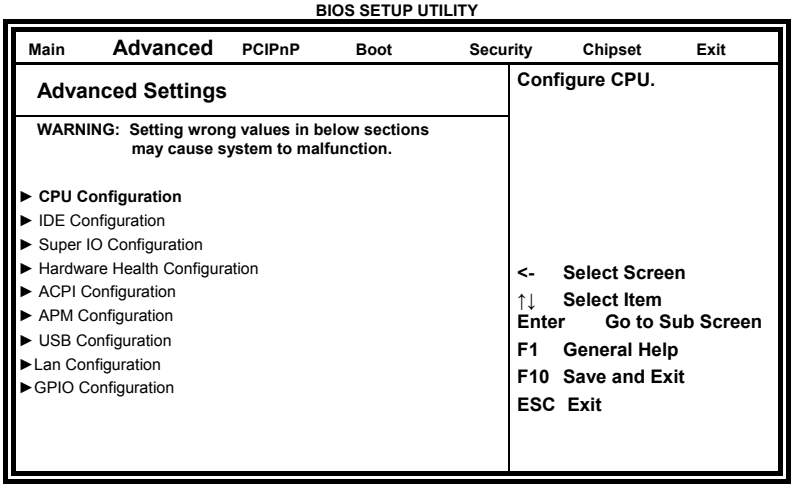


***Note**: If the system cannot boot after making and saving system changes with Setup, the AMI BIOS supports an override to the CMOS settings that resets your system to its default.*

***Warning:** It is strongly recommended that you avoid making any changes to the chipset defaults. These defaults have been carefully chosen by both AMI and your system manufacturer to provide the absolute maximum performance and reliability. Changing the defaults could cause the system to become unstable and crash in some cases.*

Advanced Settings

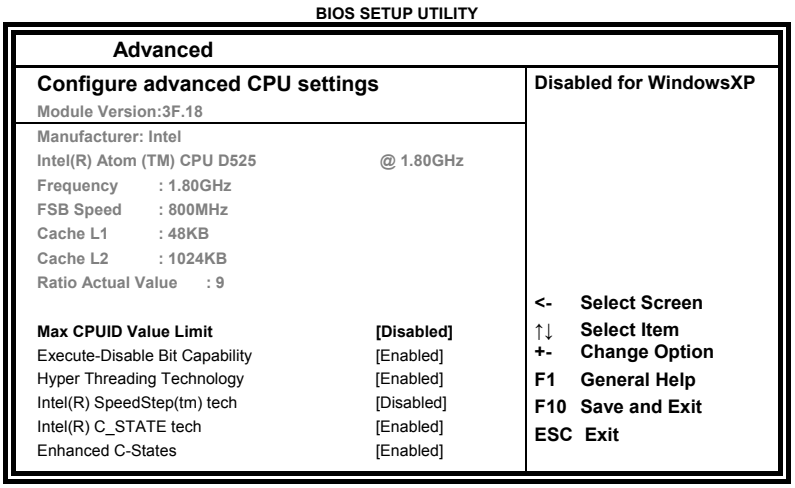
This section allows you to configure and improve your system and allows you to set up some system features according to your preference.



The Advanced BIOS Settings contains the following sections:

► CPU Configurations
► IDE Configuration
► Super IO Configuration
► Hardware Health Configuration
► ACPI Configuration
► APM Configuration
► USB Configuration
► LAN Configuration
► GPIO Configuration

The fields in each section are shown in the following pages, as seen in the computer screen. Please note that setting the wrong values may cause the system to malfunction. If unsure, please contact technical support of your supplier.



The CPU Configuration menu shows the following CPU details:
Manufacturer: the name of the CPU manufacturer
Brand String: the brand name of the CPU being used
Frequency: the CPU processing speed
FSB Speed: the FSB speed
Cache L1: the CPU L1 cache size; Cache L2: the CPU L2 cache

Max CPUID Value Limit
Disabled for WindowsXP.

Execute-Disable Bit Capability
XD can prevent certain classes of malicious buffer overflow attacks when combined with a supporting OS.

Hyper Threading Technology
Enabled for Windows XP and Linux (OS optimized for Hyper-Threading Technology) and Disabled for other OS (OS not optimized for Hyper-Threading Technology). When Disabled, only one thread per enabled core is enabled.

Intel SpeedStep(tm) tech (Pineview-M)
Disabled: Disable GV3
Enabled: Enable GV3

Intel(R) C-STATE tech
CState:CPU idle is set to C2 C3 C4 State

Enhanced C-States
CState:CPU idle is set to Enhanced C-States.

BIOS SETUP UTILITY

Advanced		
IDE Configuration		**Options**
ATA/IDE Configuration	**[Enhanced]**	**Disabled**
Configure SATA as	[IDE]	**Compatible**
		Enhanced
► Primary IDE Master	: [Not Detected]	
► Secondary IDE Master	: [Not Detected]	
Hard Disk Write Protect	[Disabled]	**<- Select Screen**
IDE Detect Time Out (Sec)	[35]	**↑↓ Select Item**
ATA(PI) 80Pin Cable Detection	[Host & Device]	**+- Change Option**
		F1 General Help
		F10 Save and Exit
		ESC Exit

The IDE Configuration menu is used to change and/or set the configuration of the IDE devices installed in the system.

ATA/IDE Configuration

(1) Disabled.
(2) Compatible.
(3) Enhanced

Configure SATA as

(1) IDE Mode.
(2) AHCI Mode.

BIOS SETUP UTILITY

Advanced		
Configure Win627UHG Super IO Chipset		**Allows BIOS to Select Serial Port Base Addresses**
Serial Port1 Address	[3F8/IRQ4]	
Device Mode	[RS232 Enable]	
Serial Port2 Address	[2F8/IRQ3]	
Serial Port2 Mode	[Normal]	**<- Select Screen**
Serial Port3 Address	[3E8]	**↑↓ Select Item**
Serial PortC IRQ	[IRQ11]	**+- Change Option**
Serial Port4 Address	[2E8]	**F1 General Help**
Serial PortD IRQ	[IRQ10]	**F10 Save and Exit**
Restore on AC Power Loss	[Power Off]	**ESC Exit**

Onboard Serial Port

The default values are:

Serial Port 1: 3F8/IRQ4
Serial Port 2: 2F8/IRQ3
Serial Port 3: 3E8/IRQ11
Serial Port 4: 2E8/IRQ10

Restore on AC Power Loss

This field sets the system power status whether *Power On or Power Off* when power returns to the system from a power failure situation.

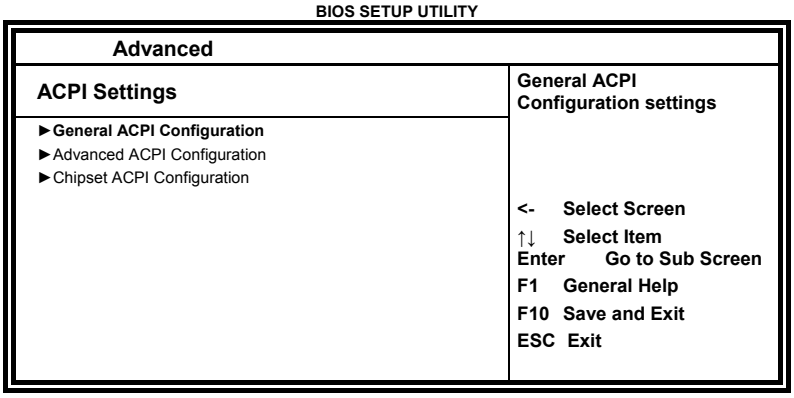
BIOS SETUP UTILITY

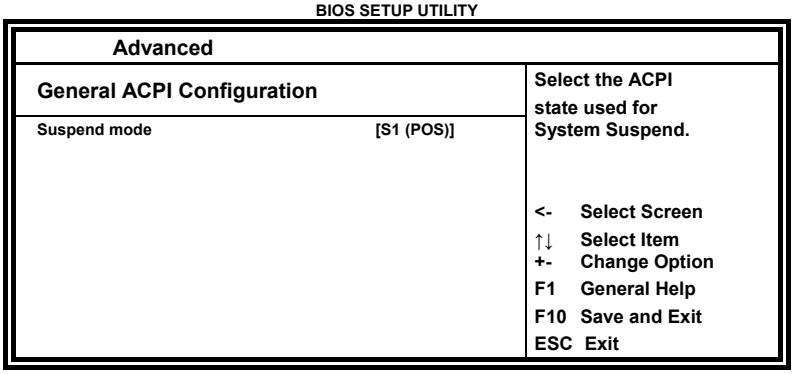
Advanced		
Hardware Health Configuration		**Options**
System Temperature	:42°C/107°F	**Disabled**
CPU Temperature	:37°C/98°F	**70°C/158°F**
CPU FAN Speed	:6750 RPM	**75°C/167°F**
		80°C/176°F
		85°C/185°F
Vcore	:1.136 V	**90°C/194°F**
12 V	:11.904 V	**95°C/203°F**
3.3V	:3.408 V	
1.5V	:1.504V	
VBAT	: 3.536V	
CPU Shutdown temperature	**[Disabled]**	**<- Select Screen**
		↑↓ Select Item
		+- Change Option
		Tab Select Field
		F1 General Help
		F10 Save and Exit
		ESC Exit

The Hardware Health Configuration menu is used to show the operating temperature, fan speeds and system voltages.

CPU Shutdown temperature

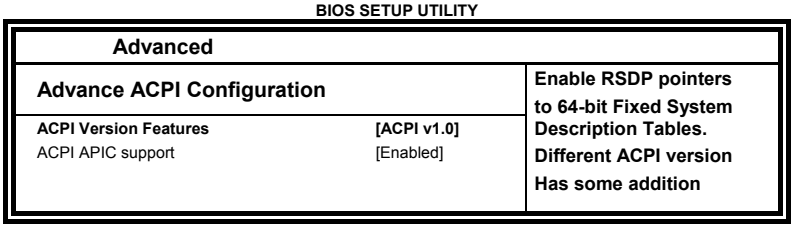
The system will shut down automatically under OS with ACPI mode, when the CPU temperature reaches the configured temperature.

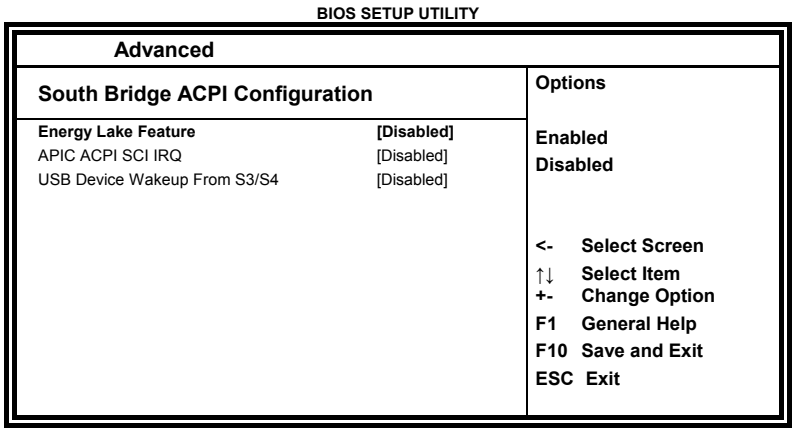


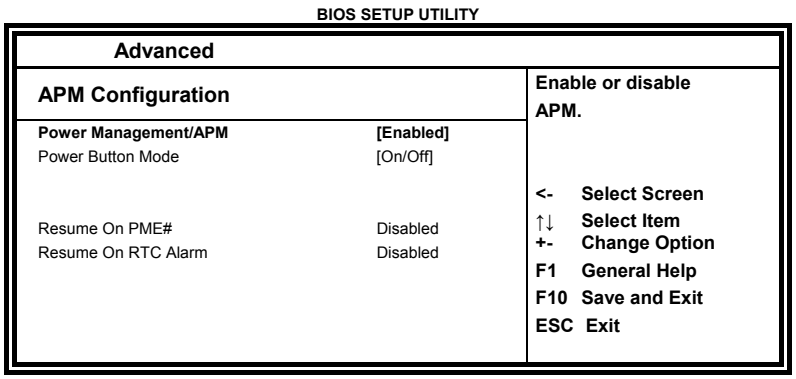


Suspend Mode

The options of this field are *S1*, *S3* and *Auto*.





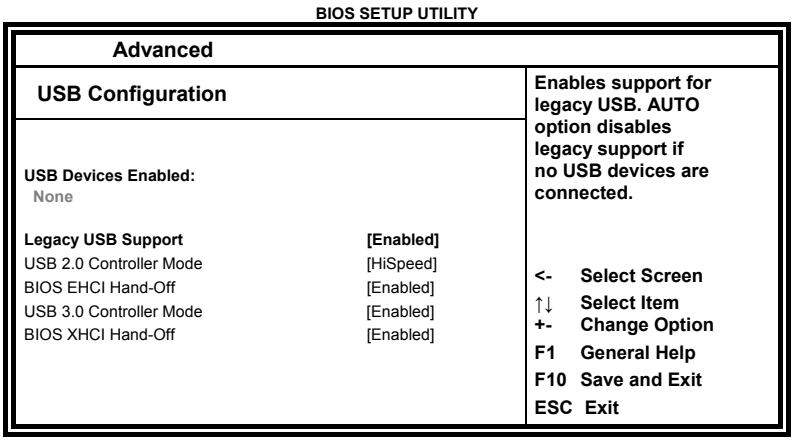


Power Management/APM
By default, this field is set to *Enabled*.

Power Button Mode
Go into On/Off, or Suspend when power button is pressed.

Resume on PME#
This option is used enable activity on the PCI PME (power management event) controller to wake up the system from a suspend or standby state

Resume On RTC Alarm
This option is used to specify the time the system should be awakened from a suspended state



The USB Configuration menu is used to read USB configuration information and configure the USB settings.

Legacy USB Support
Enables support for legacy USB. AUTO option disables legacy support if no USB devices are connected.

USB 2.0 Controller Mode
Configures the USB 2.0 controller in HiSpeed (480Mbps) or FullSpeed (12Mbps).This option is enabled by HiSpeed.

BIOS EHCI Hand-Off
Enabled/Disabled. This is a workaround for Oses without EHCI hand-off support. The EHCI ownership change should be claimed by EHCI driver.

USB 3.0 Controller Mode
Enabled USB3.0 legacy support

BIOS XHCI Hand-Off
This is a workaround for OSes without XHCI hand-off support. The XHCI ownership change should claim by XHCI driver.

Advanced		
LAN Configuration		**Options**
Onboard LAN Option ROM	[Disabled]	**Disabled** **Enabled** **<- Select Screen** **↑↓ Select Item** **+- Change Field** **F1 General Help** **F10 Save and Exit** **ESC Exit**

Onboard LAN Option ROM
Enabled/Disabled. Onboard LAN option ROM.

Advanced		
GPIO Configuration		**Options**
GPIO12	[high]	**Disabled** **Enabled** **<- Select Screen** **↑↓ Select Item** **+- Change Field** **F1 General Help** **F10 Save and Exit** **ESC Exit**
GPIO17	[Low]	
GPIO18	[high]	

PCIPnP Settings

This option configures the PCI/PnP settings.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

Advanced PCI/PnP Settings

WARNING: Setting wrong values in below sections may cause system to malfunction.

Clear NVRAM	[No]
Plug & Play O/S	[No]
PCI Latency Timer	[64]
Allocate IRQ to PCI VGA	[Yes]
Palette Snooping	[Disabled]
PCI IDE BusMaster	[Disabled]
OffBoard PCI/ISA IDE Card	[Auto]
IRQ3	[Available]
IRQ4	[Available]
IRQ5	[Available]
IRQ7	[Available]
IRQ9	[Available]
IRQ10	[Available]
IRQ11	[Available]
IRQ14	[Available]
IRQ15	[Available]
DMA Channel 0	[Available]
DMA Channel 1	[Available]
DMA Channel 3	[Available]
DMA Channel 5	[Available]
DMA Channel 6	[Available]
DMA Channel 7	[Available]
Reserved Memory Size	[Disabled]

NO: lets the BIOS Configure all the Devices in the system. YES: lets the operating system configure Plug and Play (PnP) devices not required for boot if your system has a Plug and Play operating system.

<- Select Screen
↑↓ Select Item
+- Change Option
F1 General Help
F10 Save and Exit
ESC Exit

Plug & Play O/S

This lets BIOS configure all devices in the system or lets the OS configure PnP devices not required for boot if your system has a Plug and Play OS.

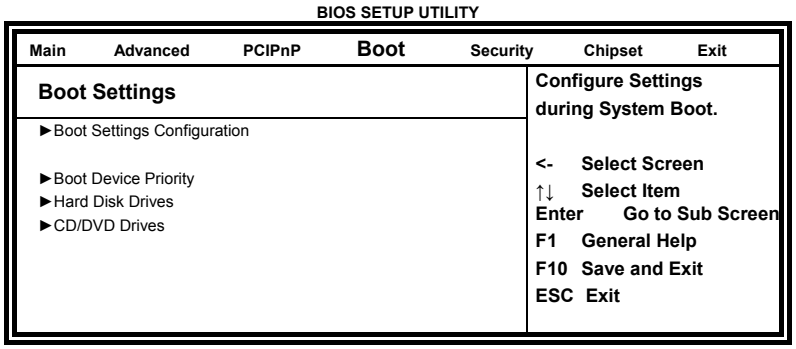
Allocate IRQ to PCI VGA

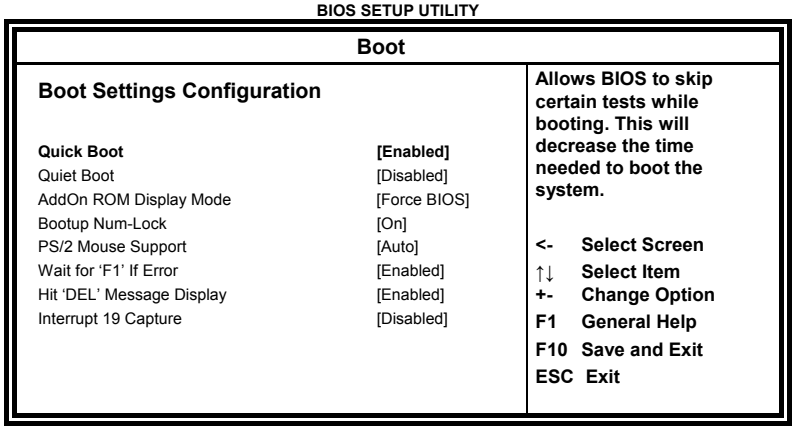
This assigns IRQ to PCI VGA card if card requests IRQ or doesn't assign IRQ to PCI VGA card even if card requests an IRQ.

IRQ#

Use the IRQ# address to specify what IRQs can be assigned to a particular peripheral device.

Boot Settings





Quick Boot

This allows BIOS to skip certain tests while booting. This will decrease the time needed to boot the system.

Quite Boot

When disabled, this displays normal POST messages. When enabled, this displays OEM Logo instead of POST messages.

AddOn ROM Display Mode

This allows user to force BIOS/Option ROM of add-on cards to be displayed during quiet boot.

Bootup Num-Lock
This select the power-on state for numlock.

PS/2 Mouse Support
This select support for PS/2 mouse.

Wait for 'F1' If Error
When set to Enabled, the system waits for the F1 key to be pressed when error occurs. This allows option ROM to trap interrupt 19.

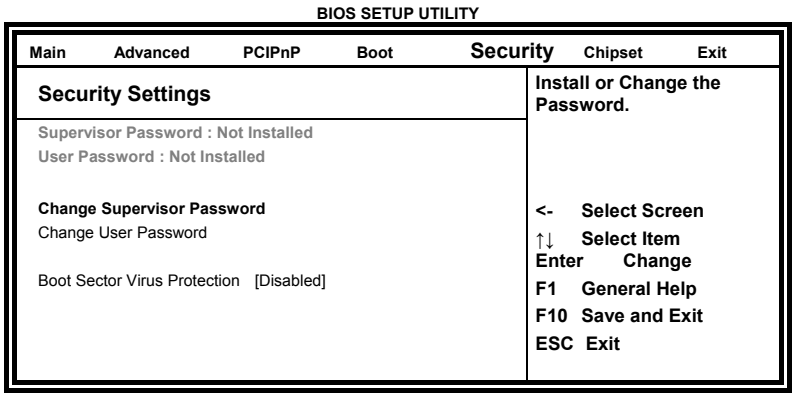
Hit <DEL> Message Display
This displays "Press <DEL> to run Setup" in POST.

Interrupt 19 Capture
This allows option ROMs to trap interrupt 19.

Security Settings

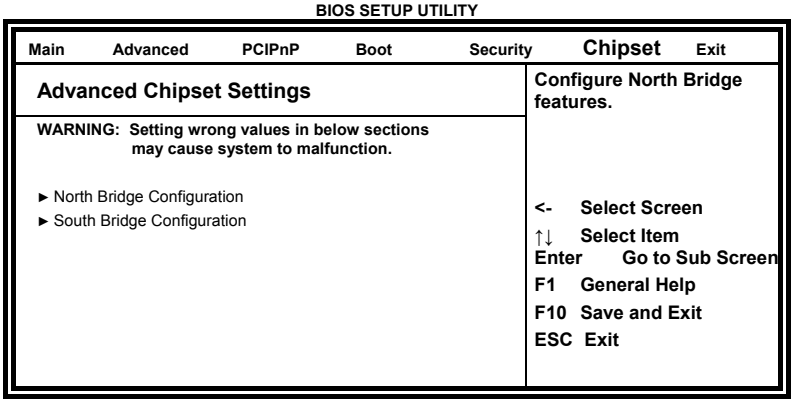
This setting comes with two options set the system password. Supervisor Password sets a password that will be used to protect the system and Setup utility. User Password sets a password that will be used exclusively on the system. To specify a password, highlight the type you want and press <Enter>. The Enter Password: message prompts on the screen. Type the password and press <Enter>. The system confirms your password by asking you to type it again. After setting a password, the screen automatically returns to the main screen.

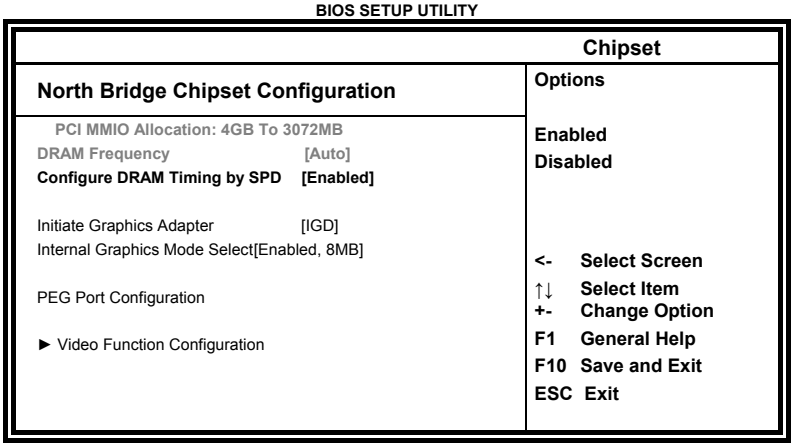
To disable a password, just press the <Enter> key when you are prompted to enter the password. A message will confirm the password to be disabled. Once the password is disabled, the system will boot and you can enter Setup freely.



Advanced Chipset Settings

This setting configures the north bridge and south bridge settings. WARNING! Setting the wrong values may cause the system to malfunction.



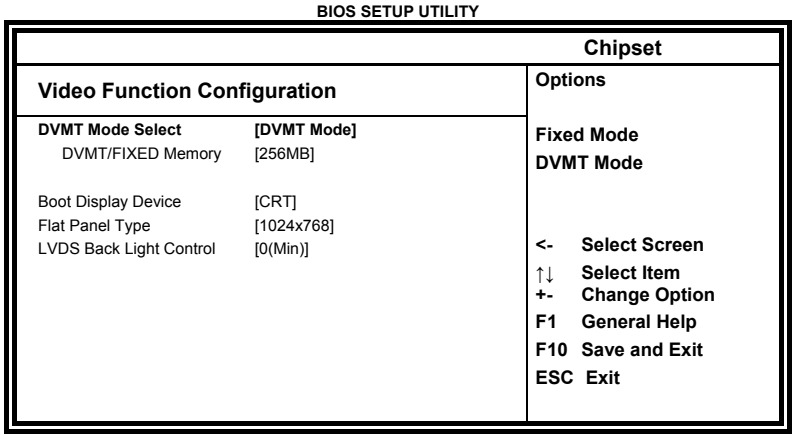


Configure DRAM Timing by SPD

When this item is enabled, the DRAM timing parameters are set according to the DRAM SPD (Serial Presence Detect). When disabled, you can manually set the DRAM timing parameters through the DRAM sub-items.

Initiate Graphic Adapter
Select which graphics controller to use as the primary boot device. This option, by default, is set to IGD.

Internal Graphics Mode Select
Use the feature to set the amount of system memory to be used by the internal graphics that requires a specified area of memory.



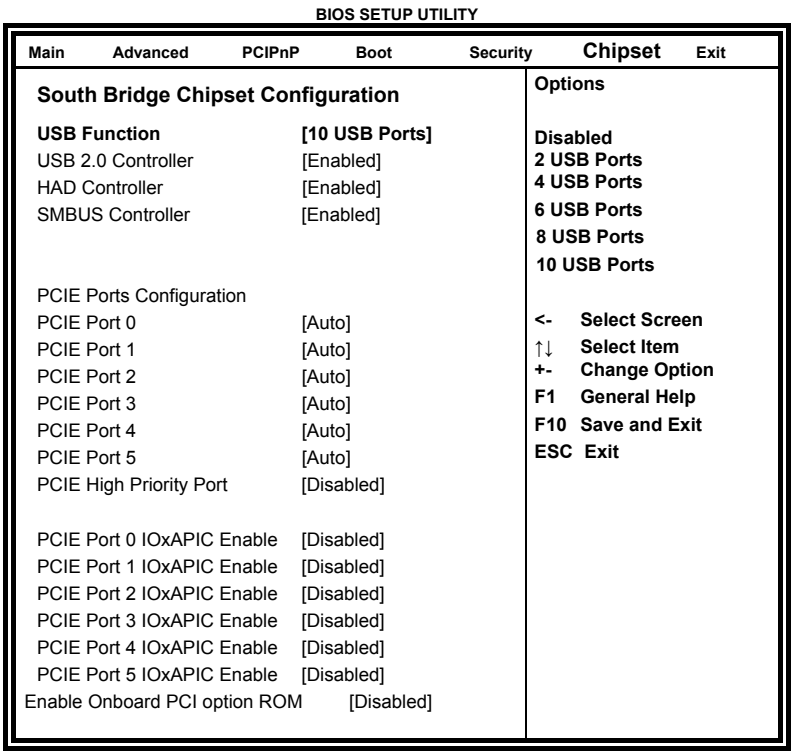
DVMT Mode Select
Select the control mode of memory built-in graphics capabilities. This option, by default, is set to DVMT Mode.

DVMT/FIXED Memory
Sets the maximum memory size assigned to the integrated graphics capabilities. This option, by default, is set to 256MB.

Boot Display Device
This option is used to select the display device used by the system when it boots.

Flat Panel Type
This option is used to select the type of flat panel connected to the system.
Options include: 640x480 / 800x600 / 1024x768 / 800x480 / 1280x768 / 1280x800 / 1280x600.

LVDS Back Light Control
Select LVDS Back Light.



USB Function
Enables the number of USB ports desired or disables the USB function.

USB 2.0 Controller
This option is disabled by default.

HDA Controller
This option is used to enable the Southbridge high definition audio controller.

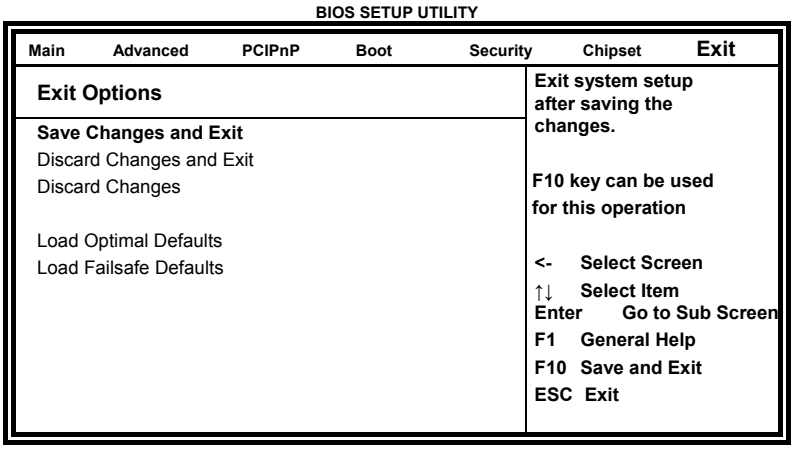
SMBUS Controller
This option is enabled by default.

Clock-Gen Spectrum
This option is disabled by default.

Enable Onboard PCI option ROM
This option is disabled by default.

Exit Setup

The exit setup has the following settings which are:



Save Changes and Exit

This option allows you to determine whether or not to accept the modifications and save all changes into the CMOS memory before exit.

Discard Changes and Exit

This option allows you to exit the Setup utility without saving the changes you have made in this session.

Discard Changes

This option allows you to discard all the changes that you have made in this session.

Load Optimal Defaults

This option allows you to load the default values to your system configuration. These default settings are optimal and enable all high performance features.

Load Failsafe Defaults

This option allows you to load the troubleshooting default values permanently stored in the BIOS ROM. These default settings are non-optimal and disable all high-performance features.

This page is intentionally left blank.

Drivers Installation

This section describes the installation procedures for software and drivers under the Windows XP, Windows Vista and Windows 7. The software and drivers are included with the motherboard. If you find the items missing, please contact the vendor where you made the purchase. The contents of this section include the following:

IMPORTANT NOTE:
After installing your Windows operating system (Windows XP/ Vista/ 7), you must install first the Intel Chipset Software Installation Utility before proceeding with the drivers installation.

Intel Chipset Software Installation Utility

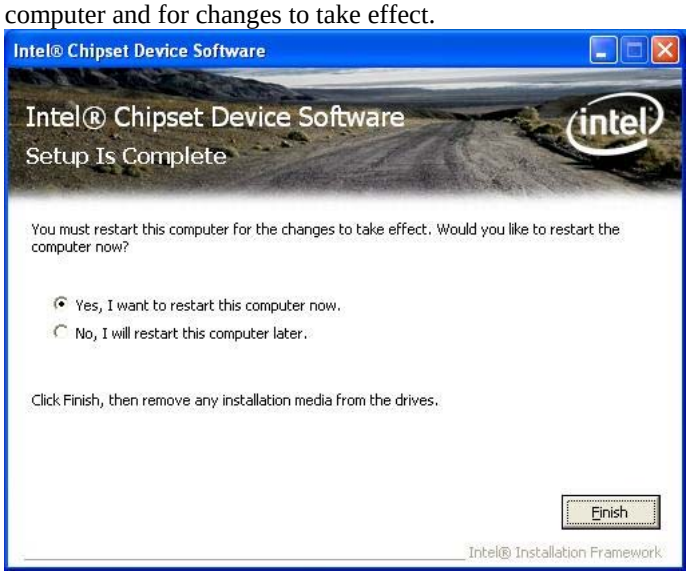
The Intel Chipset Drivers should be installed first before the software drivers to enable Plug & Play INF support for Intel chipset components. Follow the instructions below to complete the installation under Windows XP/Vista/7.

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Intel(R) Chipset Software Installation Utility***.

2. When the welcome screen to the Intel(R) Chipset Device Software appears, click ***Next*** to continue.

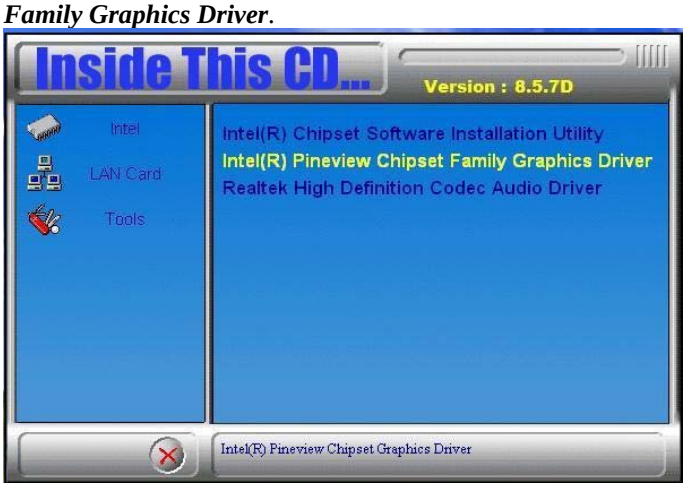
3.When the Setup Progress screen appears, click ***Next*** to continue.

4. The Setup process is now complete. Click ***Finish*** to restart the computer and for changes to take effect.



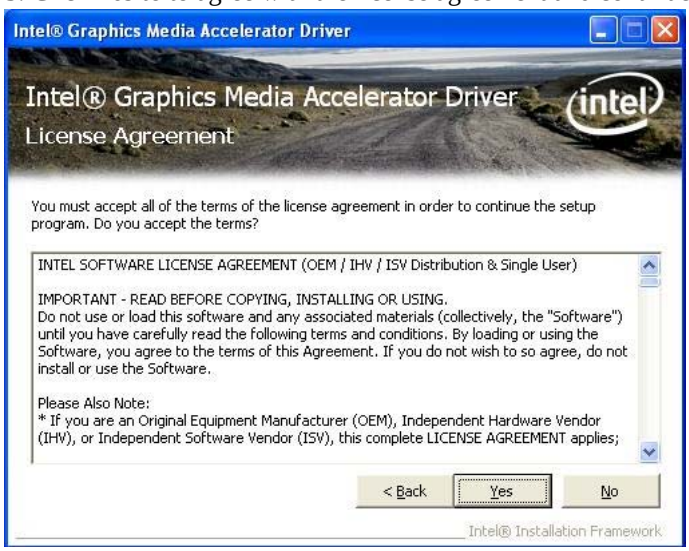
Intel Pineview Chipset Family Graphics Driver Installation

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Intel(R) Pineview Chipset Drivers***. Click ***Intel(R) Pineview Chipset Family Graphics Driver***.

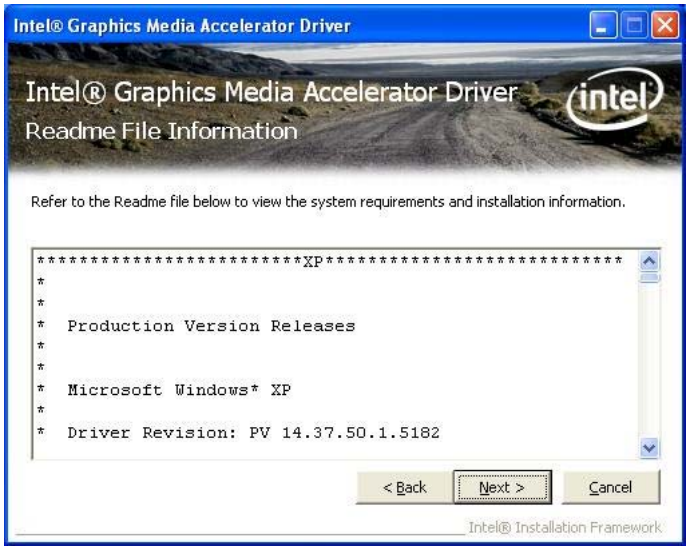


2. When the welcome screen of the Intel(R) Graphics Media Accelerator Driver appears, click ***Next*** to continue.

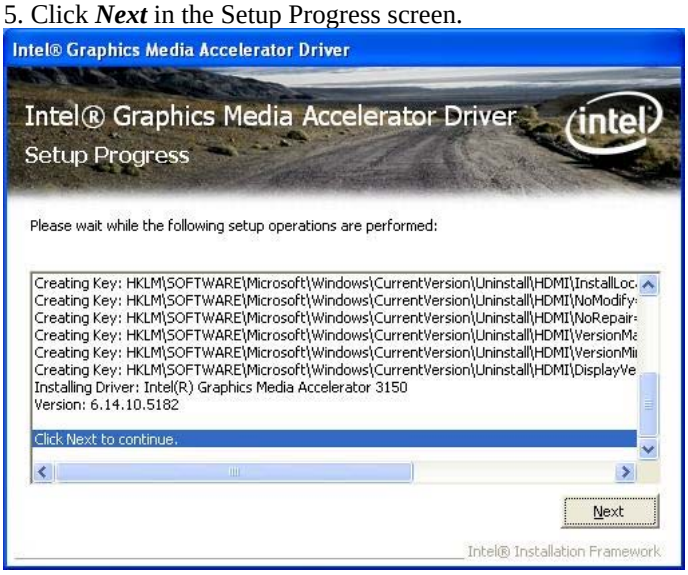
3. Click ***Yes*** to to agree with the license agreement and continue.



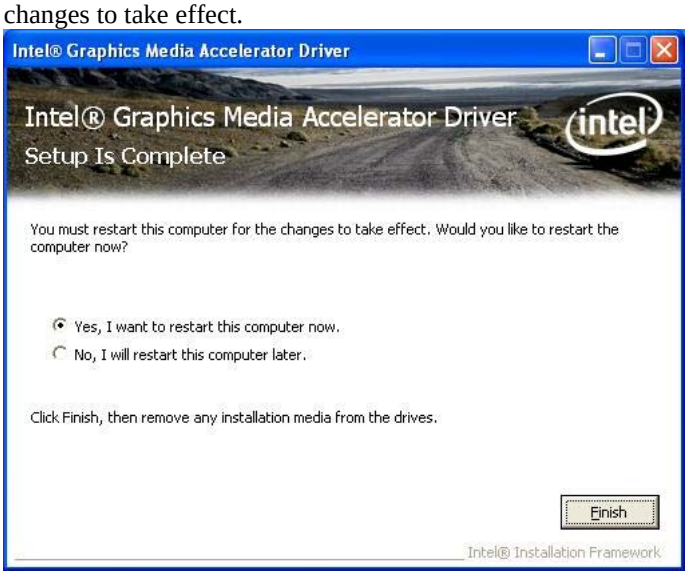
4. Click ***Next*** in the Readme File Information screen.



5. Click ***Next*** in the Setup Progress screen.

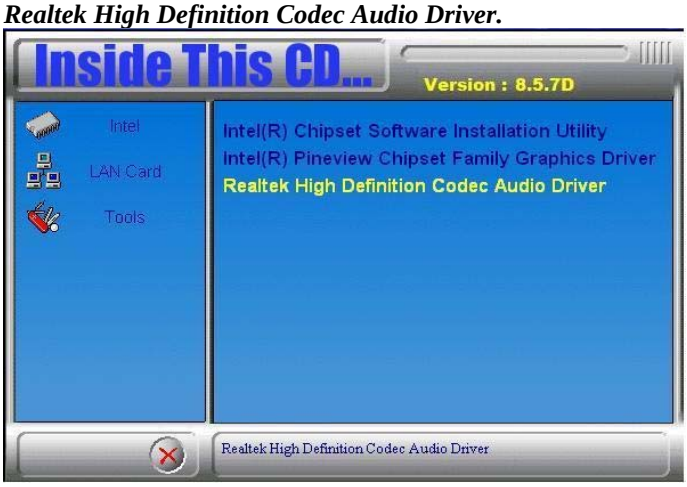


6. Setup is now complete. Click ***Finish*** to restart the computer and for changes to take effect.



Realtek High Definition Codec Audio Driver Installation

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Realtek High Definition Codec Audio Driver.***



2.When the welcome screen to InstallShield Wizard appears, click ***Next*** to start the installation.

3.When the InstallShieldWizard has finished performing maintenance operations on Realtek High Definition Codec Audio Driver, click ***Finish*** to restart the computer.

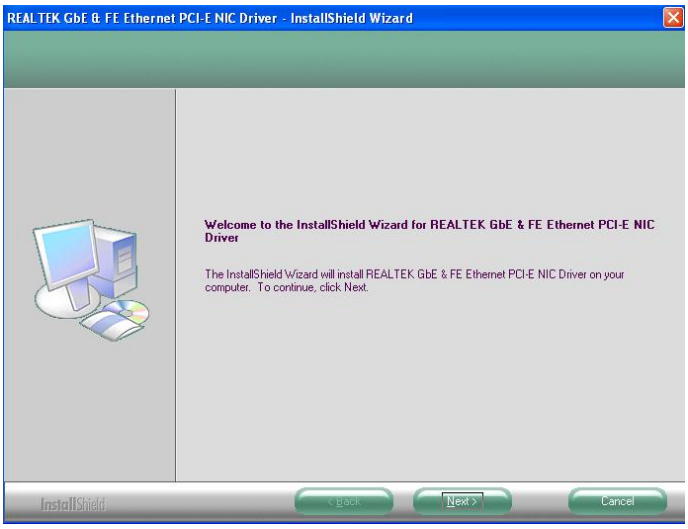
Realtek GbE_FE Ethernet PCI-E NIC Drivers Installation

Follow the steps below to install Realtek RTL8111E LAN Drivers.

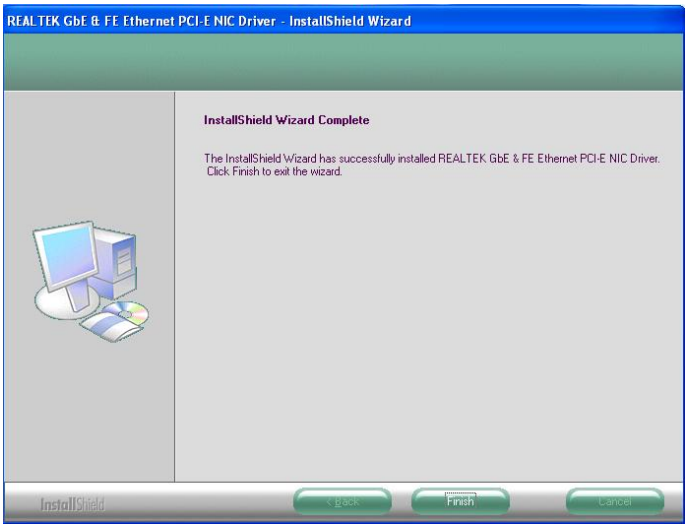
1. Insert the CD that comes with the board. Click ***LAN Card***, then ***Realtek Lan Controller Drivers***.

2. Click ***Realtek RTL8111E LAN Drivers.***

3.When the welcome screen to InstallShield Wizard appears, click ***Next*** to start the installation



4.When the InstallShieldWizard has finished installing the Realtek LAN drivers, click ***Finish***.



Appendix

A. I/O Port Address Map

Each peripheral device in the system is assigned a set of I/O port addresses that also becomes the identity of the device. The following table lists the I/O port addresses used.

Address	Device Description
000h - 01Fh	DMA Controller #1
020h - 03Fh	Interrupt Controller #1
040h - 05Fh	Timer
060h - 06Fh	Keyboard Controller
070h - 07Fh	Real Time Clock, NMI
080h - 09Fh	DMA Page Register
0A0h - 0BFh	Interrupt Controller #2
0C0h - 0DFh	DMA Controller #2
0F0h	Clear Math Coprocessor Busy Signal
0F1h	Reset Math Coprocessor
1F0h - 1F7h	IDE Interface
2B0h - 2DFh	Graphics adapter Controller
2E8h - 2EFh	Serial Port #4(COM4)
2F8h - 2FFh	Serial Port #2(COM2)
360h - 36Fh	Network Ports
3B0h - 3BFh	Monochrome & Printer adapter
3C0h - 3CFh	EGA adapter
3D0h - 3DFh	CGA adapter
3E8h - 3EFh	Serial Port #3(COM3)
3F8h - 3FFh	Serial Port #1(COM1)

B. Interrupt Request Lines (IRQ)

Peripheral devices use interrupt request lines to notify CPU for the service required. The following table shows the IRQ used by the devices on board.

Level	Function
IRQ0	System Timer Output
IRQ1	Keyboard
IRQ2	Interrupt Cascade
IRQ3	Serial Port #2
IRQ4	Serial Port #1
IRQ5	Reserved
IRQ6	Reserved
IRQ7	Reserved
IRQ8	Real Time Clock
IRQ9	Reserved
IRQ10	Serial Port #4
IRQ11	Serial Port #3
IRQ12	PS/2 Mouse
IRQ13	80287
IRQ14	Primary IDE

C. Watchdog Timer Configuration

The WDT is used to generate a variety of output signals after a user programmable count. The WDT is suitable for use in the prevention of system lock-up, such as when software becomes trapped in a deadlock. Under these sorts of circumstances, the timer will count to zero and the selected outputs will be driven. Under normal circumstance, the user will restart the WDT at regular intervals before the timer counts to zero.

SAMPLE CODE:

```
File of the W627UHG.CPP
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include "W627UHG.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int W627UHG_BASE;
void Unlock_W627UHG (void);
void Lock_W627UHG (void);
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void)
{
        unsigned int result;
        unsigned char ucDid;

        W627UHG_BASE = 0x4E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x2E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x00;
        result = W627UHG_BASE;

Init_Finish:
        return (result);
}
//---------------------------------------------------------------------------
void Unlock_W627UHG (void)
```

```
{
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
}
//-----------------------------------------------------------------------------
void Lock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_LOCK);
}
//-----------------------------------------------------------------------------
void Set_W627UHG_LD( unsigned char LD)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, W627UHG_REG_LD);
        outportb(W627UHG_DATA_PORT, LD);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
void Set_W627UHG_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        outportb(W627UHG_DATA_PORT, DATA);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
unsigned char Get_W627UHG_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        Result = inportb(W627UHG_DATA_PORT);
        Lock_W627UHG();
        return Result;
}
//-----------------------------------------------------------------------------
```

File of the W627UHG.H

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#ifndef __W627UHG_H
#define __W627UHG_H                1
//---------------------------------------------------------------------------
#define       W627UHG_INDEX_PORT            (W627UHG_BASE)
#define       W627UHG_DATA_PORT             (W627UHG_BASE+1)
//---------------------------------------------------------------------------
#define       W627UHG_REG_LD                0x07
//---------------------------------------------------------------------------
#define W627UHG_UNLOCK              0x87
#define       W627UHG_LOCK                  0xAA
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void);
void Set_W627UHG_LD( unsigned char);
void Set_W627UHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627UHG_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif        //__W627UHG_H
```

File of the MAIN.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "W627UHG.H"
//---------------------------------------------------------------------------
int main (void);

void WDTInitial(void);
void WDTEnable(unsigned char);
void WDTDisable(void);

//---------------------------------------------------------------------------
int main (void)
{
        char SIO;

        SIO = Init_W627UHG();
        if (SIO == 0)
        {
        ..... printf("Can not detect Winbond 83627UHG, program abort.\n");
        ..... return(1);
        }

        WDTInitial();

        WDTEnable(10);

        WDTDisable();

        return 0;
}
//---------------------------------------------------------------------------
void WDTInitial(void)
{
        unsigned char bBuf;
        Set_W627UHG_LD(0x08);..... //switch to logic device 8
        bBuf = Get_W627UHG_Reg(0x30);
        bBuf &= (~0x01);
        Set_W627UHG_Reg(0x30, bBuf);..... //Enable WDTO
}
//---------------------------------------------------------------------------
void WDTEnable(unsigned char NewInterval)
{
        unsigned char bBuf;

        Set_W627UHG_LD(0x08);.....
        Set_W627UHG_Reg(0x30, 0x01); ..... //enable timer
```

```
        bBuf = Get_W627UHG_Reg(0xF5);
        bBuf &= (~0x08);
        Set_W627UHG_Reg(0xF5, bBuf);.....//count mode is second

        Set_W627UHG_Reg(0xF6, NewInterval);.....//set timer
}
//-------------------------------------------------------------------------
void WDTDisable(void)
{
        Set_W627UHG_LD(0x08); .....
        Set_W627UHG_Reg(0xF6, 0x00);.....///clear watchdog timer
        Set_W627UHG_Reg(0x30, 0x00);.....
}
//-------------------------------------------------------------------------
```

D. Digital I/O Sample Code

File of the W627UHG.H

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#ifndef __W627UHG_H
#define __W627UHG_H                1
//---------------------------------------------------------------------------
#define     W627UHG_INDEX_PORT          (W627UHG_BASE)
#define     W627UHG_DATA_PORT           (W627UHG_BASE+1)
//---------------------------------------------------------------------------
#define     W627UHG_REG_LD              0x07
//---------------------------------------------------------------------------
#define W627UHG_UNLOCK              0x87
#define     W627UHG_LOCK                0xAA
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void);
void Set_W627UHG_LD( unsigned char);
void Set_W627UHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627UHG_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif      //__W627UHG_H
```

File of the W627UHG.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include "W627UHG.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int W627UHG_BASE;
void Unlock_W627UHG (void);
void Lock_W627UHG (void);
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void)
{
        unsigned int result;
        unsigned char ucDid;

        W627UHG_BASE = 0x4E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x2E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x00;
        result = W627UHG_BASE;

Init_Finish:
        return (result);
}
//---------------------------------------------------------------------------
void Unlock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
}
//---------------------------------------------------------------------------
void Lock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_LOCK);
}
//---------------------------------------------------------------------------
void Set_W627UHG_LD( unsigned char LD)
```

```
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, W627UHG_REG_LD);
        outportb(W627UHG_DATA_PORT, LD);
        Lock_W627UHG();
}
//----------------------------------------------------------------------------
void Set_W627UHG_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        outportb(W627UHG_DATA_PORT, DATA);
        Lock_W627UHG();
}
//----------------------------------------------------------------------------
unsigned char Get_W627UHG_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        Result = inportb(W627UHG_DATA_PORT);
        Lock_W627UHG();
        return Result;
}
//----------------------------------------------------------------------------
```

File of the MAIN.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "W627UHG.H"
//---------------------------------------------------------------------------
int main (void);

void Dio5Initial(void);
void Dio5SetOutput(unsigned char);
unsigned char Dio5GetInput(void);
void Dio5SetDirection(unsigned char);
unsigned char Dio5GetDirection(void);
//---------------------------------------------------------------------------
int main (void)
{
	char SIO;

	SIO = Init_W627UHG();
	if (SIO == 0)
	{
		printf("Can not detect Winbond 83627UHG, program abort.\n");
		return(1);
	}

	Dio5Initial();

	//for GPIO50..57
	Dio5SetDirection(0x0F);//GP50..53 = input, GP54..57=output
	printf("Current DIO direction = 0x%X\n", Dio5GetDirection());

	printf("Current DIO status = 0x%X\n", Dio5GetInput());

	printf("Set DIO output to high\n");
	Dio5SetOutput(0x0F);

	printf("Set DIO output to low\n");
	Dio5SetOutput(0x00);

	return 0;
```

```
}
//---------------------------------------------------------------------------
void Dio5Initial(void)
{
        unsigned char ucBuf;

        Set_W627UHG_LD(0x08);                                   //switch to logic device 8
        //enable the GP5 group
        ucBuf = Get_W627UHG_Reg(0x30);
        ucBuf |= 0x02;
        Set_W627UHG_Reg(0x30, ucBuf);
}
//---------------------------------------------------------------------------
void Dio5SetOutput(unsigned char NewData)
{
        Set_W627UHG_LD(0x08);                                   //switch to logic device 8
        Set_W627UHG_Reg(0xE1, NewData);
}
//---------------------------------------------------------------------------
unsigned char Dio5GetInput(void)
{
        unsigned char result;

        Set_W627UHG_LD(0x08);                                   //switch to logic device 8
        result = Get_W627UHG_Reg(0xE1);
        return (result);
}
//---------------------------------------------------------------------------
void Dio5SetDirection(unsigned char NewData)
{
        //NewData : 1 for input, 0 for output
        Set_W627UHG_LD(0x08);                                   //switch to logic device 8
        Set_W627UHG_Reg(0xE0, NewData);
}
//---------------------------------------------------------------------------
unsigned char Dio5GetDirection(void)
{
        unsigned char result;

        Set_W627UHG_LD(0x08);                                   //switch to logic device 8
        result = Get_W627UHG_Reg(0xE0);
        return (result);
}
//---------------------------------------------------------------------------
```